

Figure 1 consists of 11 subplots, labeled (a) through (k), showing the evolution of various parameters over time (0 to 10000) for a 1000-unit network. The plots are arranged in a single column. The x-axis for all plots is 'Time' (0 to 10000). The y-axis for all plots is 'Average firing rate' (0.0 to 1.0). The plots show the following data series: (a) Average firing rate (0.0 to 1.0), (b) Average firing rate per neuron (0.0 to 1.0), (c) Average firing rate per neuron (normalized) (0.0 to 1.0), (d) Average firing rate per neuron (normalized) (0.0 to 1.0), (e) Average firing rate per neuron (normalized) (0.0 to 1.0), (f) Average firing rate per neuron (normalized) (0.0 to 1.0), (g) Average firing rate per neuron (normalized) (0.0 to 1.0), (h) Average firing rate per neuron (normalized) (0.0 to 1.0), (i) Average firing rate per neuron (normalized) (0.0 to 1.0), (j) Average firing rate per neuron (normalized) (0.0 to 1.0), and (k) Average firing rate per neuron (normalized) (0.0 to 1.0).

Diagram of a computer system 100. The system includes a monitor 102, a system unit 110, a keyboard 104, and a mouse 106. The monitor 102 is connected to the system unit 110. The system unit 110 is connected to the keyboard 104 and the mouse 106 via cables.

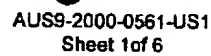


Figure 3

AUS9-2000-0561-US1

Sheet 2 of 6

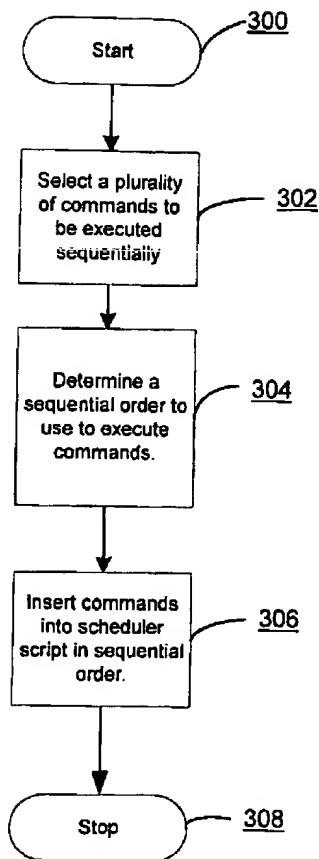
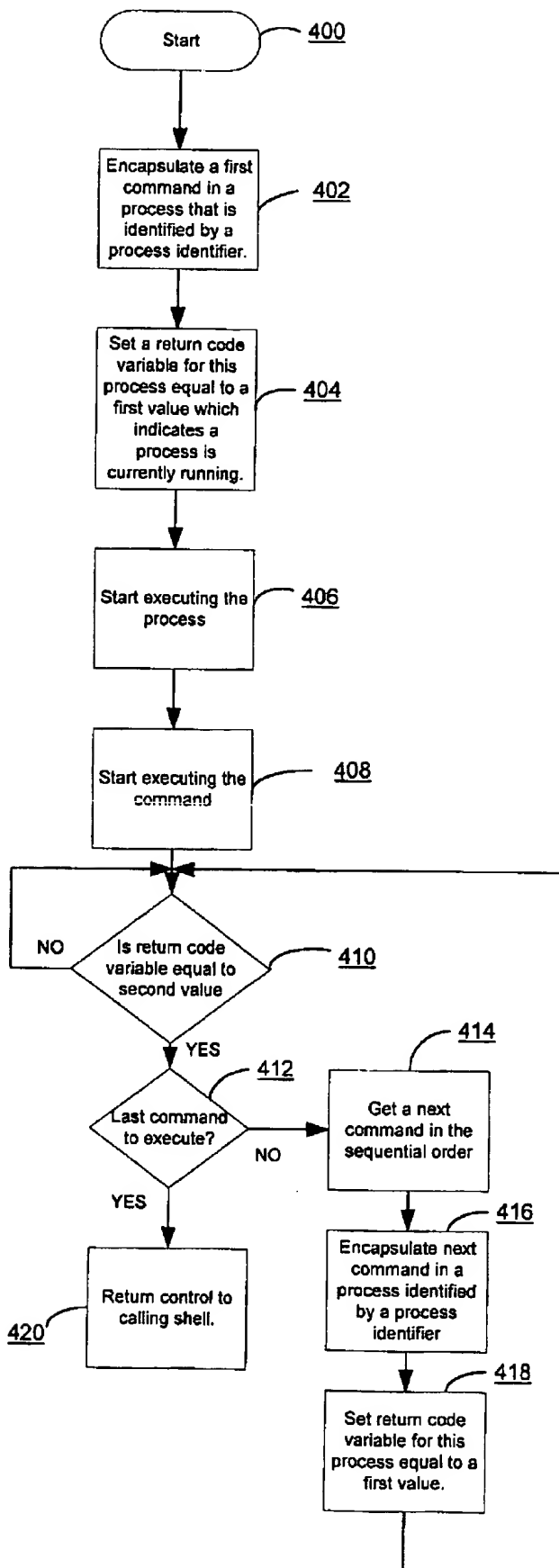


Figure 4

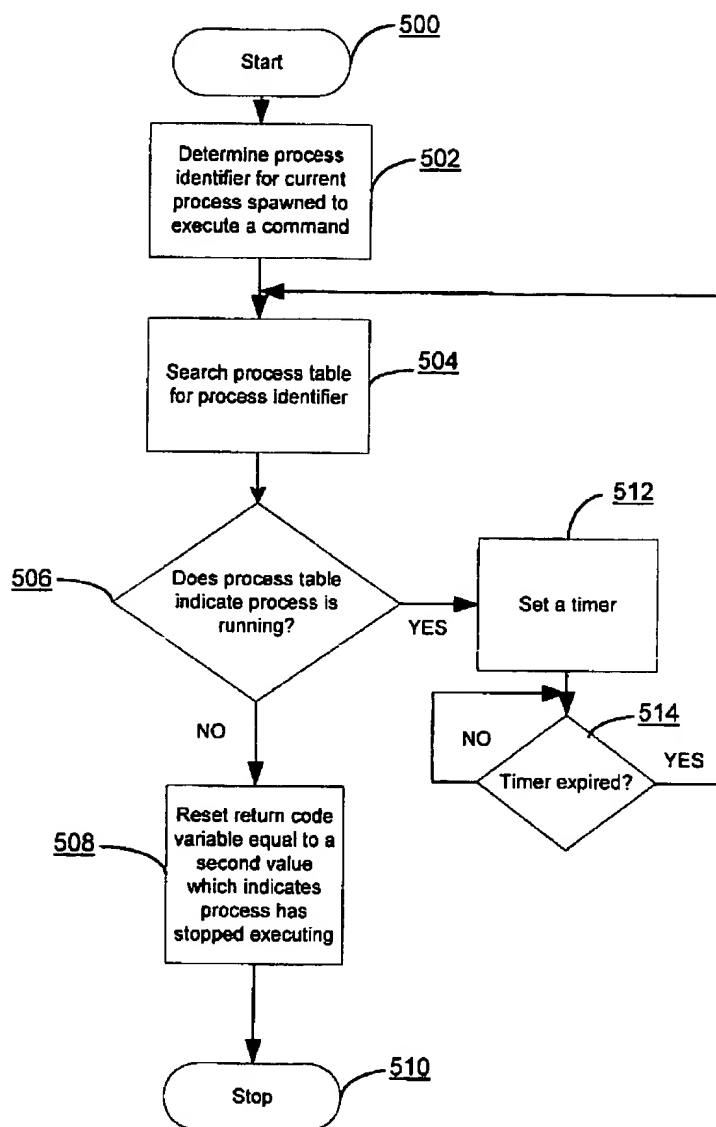
AUS9-2000-0561-US1
Sheet 3 of 6



00607F3607460

AUS9-2000-0561-US1
Sheet 4 of 6

AUS9-2000-0561-US1
Sheet 4 of 6



```
y='date +%Y%m%d'
x='date +%OH%OM%OS'
```

```
#
date >> /tmp/scheduler.log
echo "Starting scheduler" >>/tmp/scheduler.log
#-----
```

```
date >> /tmp/scheduler.log
echo "Starting process # 1 " >>/tmp/scheduler.log
rc1=0
#insert command here for process #1
while [[ $rc1 = 0 ]]
do
ps -ef|grep process#1|grep -v grep
if [[ $? -ne 0 ]]
then rc1=1
else
sleep 600
fi
done
```

```
date >> /tmp/scheduler.log
echo "Starting process # 2 " >>/tmp/scheduler.log
rc2=0
#insert command here for process #2
while [[ $rc2 = 0 ]]
do
ps -ef|grep process#2|grep -v grep
if [[ $? -ne 0 ]]
then rc2=1
else
sleep 120
fi
done
```

```
date >> /tmp/scheduler.log
echo "Starting process # 3 " >>/tmp/scheduler.log
rc3=0
#insert command here for process #3
while [[ $rc3 = 0 ]]
do
ps -ef|grep process#3|grep -v grep
if [[ $? -ne 0 ]]
then rc3=1
else
sleep 120
fi
done
```

Fig. 6
AUS9-2000-0561-US1
Sheet 5 of 6

```
#
#Example using this script with Tivoli Storage Manager (TSM)#
#
#echo "Starting DRMPHase1" >>/tmp/drm.log
#-----
#RUN DRMPHASE1 TO BACKUP STG, DB, DEVCONFIG, VOLHIST, ETC.

date >> /tmp/drm.log
echo "verifying that no node sessions are running" >>/tmp/drm.log
rc1=0
while [[ $rc1 = 0 ]]
do
dsmadm -id=admin -password=admin "select session_id,session_type,client_name from sessions
where SESSION_TYPE='Node' and CLIENT_NAME!='ENTBKDAL1'">/dev/null

if [[ $? -ne 0 ]]
then rc1=1
else
sleep 600
fi
done

date >> /tmp/drm.log
echo "Starting expire inventory" >>/tmp/drm.log
rc2=0
dsmadm -id=admin -password=admin 'expire inventory'
while [[ $rc2 = 0 ]]
do
dsmadm -id=admin -password=admin "select process_num,process from processes where
PROCESS='Expiration'">/dev/null
if [[ $? -ne 0 ]]
then rc2=1
else
sleep 120
fi
done
```

Fig. 7

AUS9-2000-0561-US1
Sheet 6 of 6